```
PPPPPPPPPPPP      LLL                IIIIIIIII    RRRRRRRRRRRR      TTTTTTTTTTTTTTT   LLL
PPPPPPPPPPPPP     LLL                IIIIIIIII    RRRRRRRRRRRRR     TTTTTTTTTTTTTTT   LLL
PPPPPPPPPPPPP     LLL                IIIIIIIII    RRRRRRRRRRRRR     TTTTTTTTTTTTTTT   LLL
PPP          PPP  LLL                   III       RRR         RRR       TTT           LLL
PPP          PPP  LLL                   III       RRR         RRR       TTT           LLL
PPP          PPP  LLL                   III       RRR         RRR       TTT           LLL
PPP          PPP  LLL                   III       RRR         RRR       TTT           LLL
PPP          PPP  LLL                   III       RRR         RRR       TTT           LLL
PPP          PPP  LLL                   III       RRR         RRR       TTT           LLL
PPPPPPPPPPPPP     LLL                   III       RRRRRRRRRRRRR         TTT           LLL
PPPPPPPPPPPPP     LLL                   III       RRRRRRRRRRRRR         TTT           LLL
PPPPPPPPPPPPP     LLL                   III       RRRRRRRRRRRRR         TTT           LLL
PPP               LLL                   III       RRR   RRR             TTT           LLL
PPP               LLL                   III       RRR   RRR             TTT           LLL
PPP               LLL                   III       RRR   RRR             TTT           LLL
PPP               LLL                   III       RRR      RRR          TTT           LLL
PPP               LLL                   III       RRR      RRR          TTT           LLL
PPP               LLL                   III       RRR      RRR          TTT           LLL
PPP               LLLLLLLLLLLLLLL    IIIIIIIII    RRR         RRR       TTT           LLLLLLLLLLLLLLL
PPP               LLLLLLLLLLLLLLL    IIIIIIIII    RRR         RRR       TTT           LLLLLLLLLLLLLLL
PPP               LLLLLLLLLLLLLLL    IIIIIIIII    RRR         RRR       TTT           LLLLLLLLLLLLLLL
```

FILEID**PLISTRING

```
PPPPPPPP    LL          IIIIII      SSSSSSSS  TTTTTTTTTT  RRRRRRRR      IIIIII    NN        NN    GGGGGGGG
PPPPPPPP    LL          IIIIII      SSSSSSSS  TTTTTTTTTT  RRRRRRRR      IIIIII    NN        NN    GGGGGGGG
PP      PP  LL            II      SS              TT      RR        RR    II      NN        NN  GG
PP      PP  LL            II      SS              TT      RR        RR    II      NN        NN  GG
PP      PP  LL            II      SS              TT      RR        RR    II      NNNN      NN  GG
PP      PP  LL            II      SS              TT      RR        RR    II      NNNN      NN  GG
PPPPPPPP    LL            II        SSSSSS        TT      RRRRRRRR          II      NN  NN    NN  GG
PPPPPPPP    LL            II        SSSSSS        TT      RRRRRRRR          II      NN  NN    NN  GG
PP          LL            II              SS      TT      RR  RR            II      NN    NNNN  GG    GGGGGG
PP          LL            II              SS      TT      RR  RR            II      NN    NNNN  GG    GGGGGG
PP          LL            II              SS      TT      RR    RR          II      NN      NN  GG        GG    ....
PP          LL            II              SS      TT      RR    RR          II      NN      NN  GG        GG    ....
PP          LLLLLLLLLL  IIIIII    SSSSSSSS        TT      RR        RR  IIIIII    NN      NN    GGGGGG          ....
PP          LLLLLLLLLL  IIIIII    SSSSSSSS        TT      RR        RR  IIIIII    NN      NN    GGGGGG          ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

```
0000     1          .title  pli$stringio - pl1 runtime get and put string
0000     2          .ident  /1-003/                                        ; Edit CGN1003
0000     3                                                                 ; Edit WHM1002
0000     4
0000     5 ;
0000     6 ;*****************************************************************************
0000     7 ;*                                                                           *
0000     8 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     9 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000    10 ;*  ALL RIGHTS RESERVED.                                                     *
0000    11 ;*                                                                           *
0000    12 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    13 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    14 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    15 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    16 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    17 ;*  TRANSFERRED.                                                             *
0000    18 ;*                                                                           *
0000    19 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    20 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    21 ;*  CORPORATION.                                                             *
0000    22 ;*                                                                           *
0000    23 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    24 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    25 ;*                                                                           *
0000    26 ;*                                                                           *
0000    27 ;*****************************************************************************
0000    28 ;
0000    29
0000    30 ;++
0000    31 ; facility:
0000    32 ;
0000    33 ;       VAX/VMS PL1 runtime library.
0000    34 ;
0000    35 ; abstract:
0000    36 ;
0000    37 ;       This module contains the pl1 runtime routine for initializing
0000    38 ;       the runtime system to perform a get or put string.
0000    39 ;
0000    40 ; author: c. spitz 4-oct-79
0000    41 ;
0000    42 ; modified:
0000    43 ;
0000    44 ;
0000    45 ;       1-002   Bill Matthews   29-September-1982
0000    46 ;
0000    47 ;               Invoke macros $defdat and rtshare instead of $defopr and share.
0000    48 ;
0000    49 ;       1-003   Chip Nylander   08-August-1983
0000    50 ;
0000    51 ;               Initialize the parent pointer with FP when stream block
0000    52 ;               allocated.
0000    53 ;
0000    54 ;--
0000    55
0000    56 ;
0000    57 ; external definitions
```

```
0000     58 ;
0000     59          $deffcb                         ;define file control block
0000     60          $defstk                         ;define stack frame offsets
0000     61          $defstr                         ;define stream block offsets
0000     62          $defdat                         ;define operand node data types
0000     63          $defpl1rtcons                   ;define pl1 run time constants
0000     64
0000     65 ;
0000     66 ; local definitions
0000     67 ;
0000     68
0000     69          rtshare                         ;sharable
```

```
                     0000     71 ;++
                     0000     72 ; pli$getstrng_r6 -- get elements from a string
                     0000     73 ;
                     0000     74 ; functional description:
                     0000     75 ;
                     0000     76 ; This routine initializes the runtime system to perform a GET STRING
                     0000     77 ; statement.
                     0000     78 ;
                     0000     79 ; the get statement is compiled into code that sets up the parameters to
                     0000     80 ; this routine, and then jsb's to it. this routine creates a file control
                     0000     81 ; block for the string, and initializes the buffer pointers to point to the
                     0000     82 ; string. we then allocate and initialize a string control block.
                     0000     83 ; following the jsb in the inline code, the compiler has generated code
                     0000     84 ; that stores the address and size of the next element in r0 and r1. the
                     0000     85 ; inline code then jsb's to the routine in pli$getedi or pli$getlis modules
                     0000     86 ; that processes elements of that data type.
                     0000     87 ;
                     0000     88 ;
                     0000     89 ; inputs:
                     0000     90 ;       (sp) - address of first element inline code
                     0000     91 ;       r0 - address of string (must be char var)
                     0000     92 ;       r1 - address of the compiled format(get edit), or 0 (get list)
                     0000     93 ; outputs:
                     0000     94 ;       ap - address of the file control block for this character string
                     0000     95 ;       r11 - address of the stream block for this get statement
                     0000     96 ; side effects:
                     0000     97 ;       r0-r6 are destroyed
                     0000     98 ;--
                     0000     99
                     0000    100 pli$getstrng_r6::
 54   5E   D0        0000    101         movl    sp,r4                   ;set get indicator
      02   11        0003    102         brb     stringcom               ;cont in common
                     0005    103 ;++
                     0005    104 ; pli$putstrng_r6
                     0005    105 ;
                     0005    106 ; functional description:
                     0005    107 ;
                     0005    108 ; This routine initializes the runtime system to put elements to a string.
                     0005    109 ;
                     0005    110 ; the put statement is compiled into code that sets up the parameters to
                     0005    111 ; this routine and then jsb's to it. this routine creates a file control
                     0005    112 ; block for the string, and initializes the
                     0005    113 ; inputs:
                     0005    114 ;       (sp) - address of first element inline code
                     0005    115 ;       r0 - address of string (char or vcha)
                     0005    116 ;       r1 - address of the compiled format(get edit), or 0 (get list)
                     0005    117 ;       r2 - max size of dest
                     0005    118 ;       r3 - data type of dest
                     0005    119 ; outputs:
                     0005    120 ;       ap - address of the file control block for this character string
                     0005    121 ;       r11 - address of the stream block for this get statement
                     0005    122 ; side effects:
                     0005    123 ;       r0-r6 are destroyed
                     0005    124 ;--
                     0005    125 pli$putstrng_r6::
 54   D4               0005    126         clrl    r4                      ;set put indicator
                     0007    127 stringcom:
```

```
                     56 8ED0  0007   128          popl    r6                              ;save return address
                5E   34   C2  000A   129          subl    #fcb_c_strlen,sp                ;get room for string file control block
                         0D  000D   130 ;
                         0D  000D   131 ; initialize file control block
                         0D  000D   132 ;
                5C   5E   D0  000D   133          movl    sp,ap                           ;set file control block addr
                6C   00   7D  0010   134          movq    #0,fcb_l_next(ap)               ;clr link fields
          0C AC 00820802 8F   D0  0013   135          movl    #<atr_m_opened! -               ;init file attributes
                         1B  001B   136                  atr_m_stream!atr_m_bfall! - ;
                         1B  001B   137                  atr_m_string>,fcb_l_attr(ap) ;
                54   D5  001B   138          tstl    r4                              ;get or put?
                09   13  001D   139          beql    10$                             ;if eql, its put
             0C AC   06   C8  001F   140          bisl    #atr_v_input,fcb_l_attr(ap) ;set input in fcb
                52   80   3C  0023   141          movzwl  (r0)+,r2                        ;get length of string
                1B   11  0026   142          brb     50$                             ;cont
             0C AC   05   C8  0028   143 10$:     bisl    #atr_v_output,fcb_l_attr(ap) ;set output in fcb
          0C AC 01000000 8F   CA  002C   144          bicl    #atr_m_vcha,fcb_l_attr(ap) ;assume not char var dest
                53   0A   D1  0034   145          cmpl    #dat_k_char,r3                  ;char dest?
                0A   13  0037   146          beql    50$                             ;if eql, yes, cont
          0C AC 01000000 8F   C8  0039   147          bisl    #atr_m_vcha,fcb_l_attr(ap) ;set vcha dest
                80   B5  0041   148          tstw    (r0)+                           ;point past length of vcha
             14 AC   50   D0  0043   149 50$:     movl    r0,fcb_l_buf(ap)                ;set buffer address
             1C AC   50   D0  0047   150          movl    r0,fcb_l_buf_pt(ap)             ;set buffer pointer
          18 AC   50   52   C1  004B   151          addl3   r2,r0,fcb_l_buf_end(ap) ;set buffer end
             20 AC   00   7D  0050   152          movq    #0,fcb_q_rfa(ap)                ;set rfa to 0
             28 AC   03   B0  0054   153          movw    #pli$c_version,fcb_w_revision(ap) ;set revision
                         58  0058   154          assume  <fcb_w_linesize+2> eq fcb_w_pagesize
                         58  0058   155          assume  <fcb_w_column+2> eq fcb_w_line
             2A AC   00   7D  0058   156          movq    #0,fcb_w_linesize(ap)   ;clr linesize, pagesize, col and line
             2A AC   52   B0  005C   157          movw    r2,fcb_w_linesize(ap)   ;set linesize to length of string
                32 AC   B4  0060   158          clrw    fcb_w_page(ap)                  ;clr page
                         63  0063   159 ;
                         63  0063   160 ; allocate stream block
                         63  0063   161 ;
       5E 00000C08 8F   C2  0063   162          subl    #str_c_len,sp                   ;alloc space for stream block
                5B   5E   D0  006A   163          movl    sp,r11                          ;set address of stream block
          14 AB   0408 CB   9E  006D   164          movab   str_l_stack_end(r11),str_l_fld_end(r11) ;set end of field
                         73  0073   165 ;
                         73  0073   166 ; initialize format stack
                         73  0073   167 ;
             08 AB   5D   D0  0073   168          movl    fp,str_l_parent(r11)    ;set default parent pointer
             04 AB   51   D0  0077   169          movl    r1,str_l_fp(r11)                ;set address of format pointer
          0C04 CB   51   D0  007B   170          movl    r1,str_l_stack(r11)     ;copy format pointer to stack
          6B 0C04 CB   9E  0080   171          movab   str_l_stack(r11),str_l_sp(r11) ;store format stack pointer
                         85  0085   172 ;
                         85  0085   173 ; return to inline code
                         85  0085   174 ;
                66   17  0085   175          jmp     (r6)                            ;return
                         87  0087   176
                         87  0087   177          .end
```

```
ATR_M_BFALL     = 00020000
ATR_M_OPENED    = 00000002
ATR_M_STREAM    = 00000800
ATR_M_STRING    = 00800000
ATR_M_VCHA      = 01000000
ATR_V_INPUT     = 00000006
ATR_V_OUTPUT    = 00000005
DAT_K_CHAR      = 0000000A
FCB_B_ENVIR       000001C2
FCB_B_ESA         0000012E
FCB_B_EXTRA       0000003D
FCB_B_FAB         000000A6
FCB_B_IDENT       00000040
FCB_B_IDENT_NAM   00000042
FCB_B_NAM         000000F6
FCB_B_NUMKCBS     0000003C
FCB_B_RAB         00000062
FCB_C_LEN         000001C2
FCB_C_STRLEN      00000034
FCB_L_ATTR        0000000C
FCB_L_BUF         00000014
FCB_L_BUF_END     00000018
FCB_L_BUF_PT      0000001C
FCB_L_CNDADDR     000001B2
FCB_L_CONDIT      000001AE
FCB_L_DTTR        00000010
FCB_L_ERROR       00000008
FCB_L_KCB         00000038
FCB_L_NEXT        00000000
FCB_L_PREVIOUS    00000004
FCB_L_PRN         00000034
FCB_Q_RFA         00000020
FCB_W_COLUMN      0000002E
FCB_W_IDENT_LEN   00000040
FCB_W_LINE        00000030
FCB_W_LINESIZE    0000002A
FCB_W_PAGE        00000032
FCB_W_PAGESIZE    0000002C
FCB_W_REVISION    00000028
PLI$C_VERSION   = 00000003
PLI$GETSTRNG_R6   00000000 RG     02
PLI$PUTSTRNG_R6   00000005 RG     02
SIZ...          = 00000001
STK_L_AP          00000008
STK_L_ARG_LIST    FFFFFFF8
STK_L_CND_HND     00000000
STK_L_CND_LST     FFFFFFF4
STK_L_DISPLAY     FFFFFFFC
STK_L_FP          0000000C
STK_L_PC          00000010
STK_L_PSL         00000004
STK_L_REGS        00000014
STRINGCOM         00000007 R      02
STR_B_FIELD       00000018
STR_C_LEN         00000C08
STR_L_FLD_END     00000014
STR_L_FLD_PT      00000010
STR_L_FP          00000004
STR_L_FS          0000000C
STR_L_PARENT      00000008
STR_L_SP          00000000
STR_L_STACK       00000C04
STR_L_STACK_END   00000408
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	FFFFFFFC (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
_PLI$CODE	00000087 (135.)	02 (2.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	13	00:00:00.10	00:00:00.38
Command processing	70	00:00:00.50	00:00:01.95
Pass 1	112	00:00:03.12	00:00:06.88
Symbol table sort	0	00:00:00.20	00:00:00.32
Pass 2	39	00:00:00.62	00:00:01.17
Symbol table output	5	00:00:00.07	00:00:00.07
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	241	00:00:04.64	00:00:10.81

The working set limit was 750 pages.
13870 bytes (28 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 185 non-local and 2 local symbols.
177 source lines were read in Pass 1, producing 10 object records in Pass 2.
16 pages of virtual memory were used to define 14 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[PLIRTL.OBJ]PLIRTMAC.MLB;1	6
_255DUA28:[SYSLIB]STARLET.MLB;2	5
TOTALS (all libraries)	11

224 GETS were required to define 11 macros.

There were no errors, warnings or information messages.

```
MACRO/ENABLE=SUPPRESSION/DISABLE=TRACEBACK/LIS=LIS$:PLISTRING/OBJ=OBJ$:PLISTRING MSRC$:PLISTRING/UPDATE=(ENH$:PLISTRING)+LIB$:PLIRTM
```

BANNER
LIS

PLIWRITE
LIS

SMBREQ
REQ

PLIVECTOR
LIS

SMBSRVSHR
MAP

FORMAT
LIS

PLIRODATA
LIS

SMBDEF
SDL

PRTSMB

DISPATCH
L13

PLISTRING
LIS

PRTSMB
MAP